BATCHED GOALS EXPLORATION FOR UNLABELED OFFLINE-TO-ONLINE RL

Anonymous authors
Paper under double-blind review

ABSTRACT

In many real-world settings, prior data is plentiful but has no reward labels for the task at hand. We study unlabeled offline-to-online RL, where an agent must use reward-free offline data to efficiently explore online. In standard RL exploration algorithms, the agent must adapt to new reward information by repeatedly updating the exploration policy with online RL objectives, which can be expensive and unstable. We propose batched goals exploration (BGE) for unlabeled offline-to-online RL, an adaptive exploration algorithm that simultaneously targets multiple novel regions in the state space without any online RL updates. BGE avoids online RL updates by decoupling behavior optimization from high-level online exploration. In particular, BGE first pre-trains a goal-conditioned agent on a prior offline dataset, then online optimizes a batch of goals such that the expected state occupancy achieved by rolling out the pre-trained goal-conditioned agent conditioned on these goals maximizes the coverage over potential high-reward states. Across challenging long-horizon, sparse-reward tasks across locomotion and manipulation, BGE is able to leverage unlabeled prior data to achieve better online sample-efficiency compared to prior methods, while only using a fraction of the computational cost online.

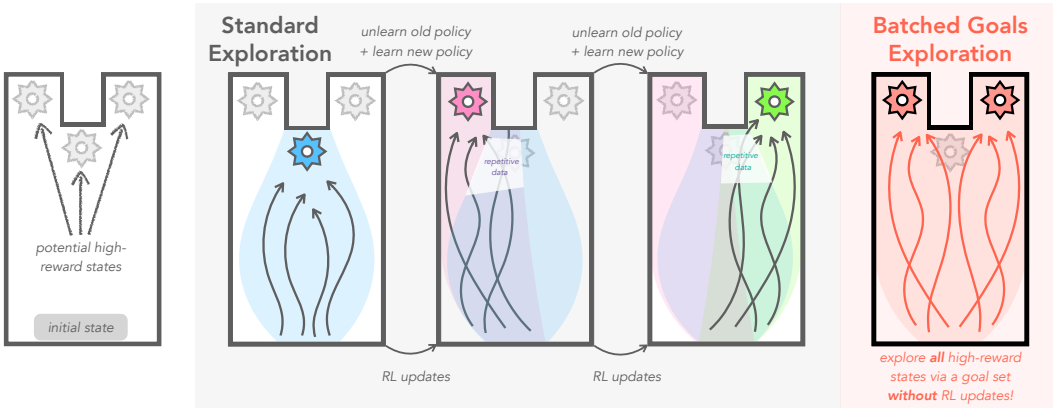


Figure 1: **Batched Goals Exploration (BGE) for unlabeled offline-to-online RL.** Standard exploration methods target one novel state at a time. Such greedy solution often results in redundant exploration (*i.e.*, overlapping shaded areas above). Instead, BGE plans a batch of goals to explore in advance such that the *combined occupancy* of the goal-directed behaviors maximally cover the novel states. To efficiently compute goal batches online, BGE pre-trains a goal-conditioned agent and its occupancy estimator on an offline dataset. This allows BGE to avoid expensive and unstable online RL updates completely.

1 INTRODUCTION

Reinforcement learning (RL) agents learning a new task face a difficult exploration problem, especially when the state space is large and rewards are sparse. In many practical settings, however, we

Reviewers: please read the Reviewer Guidelines (iclr.cc/Conferences/2027/ReviewerGuidelines) and the AI Policy for Reviewers (iclr.cc/Conferences/2027/AIPolicyForReviewers). If you used AI to expand, edit, or polish your review, please provide the input text to the LLM. Better still, consider skipping the LLM and submitting your original text: we, and the authors, are much more interested in your unedited thoughts than in what an LLM has to say. AI-assisted or not, you are putting your name and reputation behind your review: LLM-generated falsehoods, hallucinations or misrepresentations are subject to disciplinary action, which may include desk-rejecting all papers you have authored.

have access to prior data from the same environment, even if it was collected for other purposes and carries no reward labels for the task at hand. Such unlabeled data is much easier to obtain than task-specific, reward-annotated data (Li et al., 2024; Wilcoxson et al., 2025) and reveals information about the dynamics of the environment. For example, a person searching for lost keys does not need to relearn how to walk or how the rooms of their house connect; instead, they use what they already know to decide where to look (Li et al., 2024). We study this unlabeled offline-to-online RL setting, in which an agent must use a reward-free offline dataset to find and solve a new task with as few online interactions as possible.

Prior methods for this setting (Li et al., 2024; Wilcoxson et al., 2025) often rely on continual online RL updates to incorporate newly collected information to adapt the agent's exploratory behavior. This is inefficient: the agent might have a few potential hypotheses about different regions of the state space that are worth exploring based on exploration novelty signals, and when it visits one part of the state space and reduces its novelty, a standard exploration algorithm needs to *unlearn* how to go to this region and *relearn* how to visit a different region that is still underexplored. How quickly the agent can explore is thus limited by how quickly its policy and critic can adapt, and the standard way to speed up adaption is to take more gradient updates per environment step (*i.e.*, raise the update-to-data (UTD) ratio). However, high UTD training often causes the learning to be more off-policy, which can induce instability and loss of plasticity (Nikishin et al., 2023), often requiring careful regularization techniques to mitigate (D'Oro et al., 2023; Sokar et al., 2023).

In this paper, we make the observation that we can queue up a range of different exploration objectives, so that the agent can visit different parts of the state space and collect new information without even needing to change the policy. More concretely, rather than training a single RL agent that must be updated frequently online, we first pre-train a goal-conditioned agent on the prior data to produce a diverse set of behaviors and then select a subset of goal-directed behaviors to cover the regions with potentially high rewards. In this way, we avoid the difficult problem of continually molding the policy to match a changing reward belief during online exploration. Instead, we can simply select a set of goals based on the current reward belief, and collect informative data by rolling out the pre-trained goal-conditioned agent (see Figure 1). As a result, our agent can explore efficiently online, with only a fraction of the online computational cost required in prior approaches, while attaining better sample efficiency.

Our main contribution of this paper is a new exploration algorithm, **Batched Goals Exploration (BGE)** for unlabeled offline-to-online RL, that leverages offline goal-conditioned pre-training to efficiently explore online without relying on frequent policy updates. We evaluate our method in the unlabeled-offline-to-online problem setting (Section 4) on a set of challenging long-horizon, sparse-reward tasks ranging from locomotion to manipulation. Compared to prior methods, BGE consistently yields better sample efficiency.

2 Related Work

Exploration. Many online exploration techniques involve adding an exploration bonus to the true reward, then optimizing the policy with respect to this augmented reward (Stadie et al., 2015; Bellemare et al., 2016; Houthooft et al., 2017; Pathak et al., 2017; Tang et al., 2017; Ostrovski et al., 2017; Achiam & Sastry, 2017; Merel et al., 2018; Burda et al., 2018; Ermolov & Sebe, 2020; Lobel et al., 2023). The exploration bonus is often calculated to encourage the visitation of new states, diverse data, and high coverage of the state space. An orthogonal approach to accelerate exploration uses concurrent agents or ensemble methods, where multiple agents are trained simultaneously to maximize collective state entropy or guide exploration (Jung et al., 2020; Paola et al., 2025). While effective at gathering diverse data, these two classes of methods often require policy and/or critic optimization with respect to some non-stationary objectives (*e.g.*, maximizing return subject to a moving reward bonus profile). Optimizing these objectives in a stable manner often requires a lot of on-policy samples, hurting online exploration efficiency. In contrast, we use a single, pre-trained goal-conditioned agent that is frozen during online learning, completely removing the need for online policy and/or critic optimization. Instead, we direct the agent to visit novel states by adaptive commanding an appropriate set of goals obtained from a cheap optimization procedure, allowing efficient exploration without online RL updates.

Online reinforcement learning with prior data. Offline-to-online reinforcement learning (RL) methods (Lee et al., 2021; Agarwal et al., 2022; Xie et al., 2022; Song et al., 2023; Zhang et al., 2023; Zheng et al., 2023; Ball et al., 2023; Rafailov et al., 2024; Zhou et al., 2025) leverage offline data to accelerate online RL. These methods typically first use offline RL (Levine et al., 2020) to pre-train a policy using the offline data and then use this policy as the initialization for online RL fine-tuning (Rafailov et al., 2024; Wołczyk et al., 2024; Zhou et al., 2025). While these methods are designed for the setting where rewards are available in the offline dataset, there has been a growing interest in tackling the more general case in which the reward is unknown in the offline data, because oftentimes such 'unlabeled' data is much easier to acquire in real-world settings (Li et al., 2024; Hu et al., 2023; Wilcoxson et al., 2025). We refer it as the *unlabeled-offline-to-online RL* setting in this paper. Most offline-to-online RL methods are not directly applicable in this problem setting, as the offline RL pre-training phase requires reward labels. Li et al. (2024) and Wilcoxson et al. (2025) propose to leverage unlabeled prior data by learning a reward model online to label the offline data, and then using standard off-policy online RL to learn from both the labeled offline data and the online replay buffer data. While conceptually simple, the sample efficiency of these algorithms depend heavily on frequent RL updates (*e.g.*, high update-to-data ratio) so that the agent can quickly adapt to any changes in the learned reward model, and adjust its exploration behavior promptly. Such frequent updates are not only computationally inefficient but also prone to instability and Q-value overestimation (Hussing et al., 2024). Instead of quickly adapting a single agent to new data and changes in the reward model via frequent updates, our method instead pre-trains a goal-conditioned agent that is capable of producing the appropriate exploration behaviors towards any reward function. This design avoids online RL updates altogether, allowing for more efficient online exploration.

See more related work discussions on unsupervised pre-training and meta reinforcement learning in Appendix D.

3 Preliminaries

Reinforcement learning is often formalized as a Markov decision process (MDP), $\mathcal{M} = (\mathcal{S}, \mathcal{A}, \mathcal{T}, \gamma, R, \rho)$, where $\mathcal{S}$ is the set of all possible states, $\mathcal{A}$ is the set of all possible actions that a policy $\pi(a \mid s) : \mathcal{S} \to \Delta_{\mathcal{A}}$ may take, $\mathcal{T}(s' \mid s, a) : \mathcal{S} \times \mathcal{A} \to \Delta_{\mathcal{S}}$ is the transition function that describes the probability distribution over the next state s' given the current state and the action a taken at the state, γ is the discount factor, $R(s, a) : \mathcal{S} \times \mathcal{A} \to \mathbb{R}$ is the reward function, and $\rho \in \Delta_{\mathcal{S}}$ is the initial state distribution. In online RL, we design an RL agent take actions $\{a_t\}_t$ in the MDP $\mathcal{M}$ to generate trajectory rollouts with a maximum length of T: $(s_0, a_0, s_1, a_1, \cdots, s_T)$, where $s_{t+1} \sim \mathcal{T}(\cdot \mid s_t, a_t), s_0 \sim \rho$. The goal is to use as little interaction as possible to produce a policy π that maximizes its expected cumulative discounted reward over the full trajectory:

$$\eta(\pi) = \mathbb{E}_{s_{t+1} \sim \mathcal{T}(\cdot|s_t, a_t), a_t \sim \pi(\cdot|s_t), s_0 \sim \rho} \left[\sum_{t=0}^{\infty} \gamma^t r(s_t, a_t) \right]. \tag{1}$$

For long-horizon, sparse reward tasks, directly optimizing the return can be challenging because optimization landscape is flat (*i.e.*, zero return everywhere except a small subset of policies centered around the optimal policy). To ensure policy convergence, we often add a reward bonus $R_b : \mathcal{S} \to \mathbb{R}_0^+$ on top of the original reward function to construct an augmented reward function $\tilde{R}(s, a) := R(s, a) + R_b(s)$. The reward bonus function places low values on states that are frequently visited by the RL agent and high values on states that are less frequently visited. This allows the RL agent to continually seek out for novel states in the absence of reward signal, and eventually find the sparse reward signal needed to solve the task. In many cases, this augmented reward function can be seen as an approximation of the upper-confidence bound (UCB) on the reward function. Our method also adopts this reward bonus formulation to guide online exploration.

Goal-conditioned reinforcement learning (Kaelbling, 1993) builds on top of standard MDP formulation, with the introduction of an goal space $\mathcal{G}$. The objective of goal-conditioned RL (Kaelbling, 1993; Schaul et al., 2015b; Liu et al., 2022) is to learn a goal-conditioned policy $\pi(a_t \mid s_t, g) : \mathcal{S} \times \mathcal{G} \to \Delta_{\mathcal{A}}$ that maximizes the return $\eta_g(\pi)$ for all $g \in \mathcal{G}$:

$$\eta_g(\pi) = \mathbb{E}_{s_{t+1} \sim \mathcal{T}(\cdot|s_t, a_t), a_t \sim \pi(\cdot|s_t, g), s_0 \sim \rho} \left[\sum_{t=0}^{\infty} \gamma^t \mathbb{I}_{s_t = g} \right], \tag{2}$$

where the goal representation $\Psi : \mathcal{S} \to \mathcal{G}$ is a deterministic map from state to goal. Goal-conditioned RL is a popular unsupervised pre-training objective for online exploration and goal-conditioned pre-training can be done either online (Schaul et al., 2015a; Andrychowicz et al., 2018; Choi et al., 2021; Mendonca et al., 2021; Bae et al., 2024) or offline (on an offline dataset of environment trajectory rollouts) (Chebotar et al., 2021; Park et al., 2023; 2025a). As we will describe in Section 5, our method leverages an offline pre-trained goal-conditioned policy to provide a diverse set of exploration behaviors, and it allows us to transpose the online exploration problem as a goal-set optimization problem where we carefully find an appropriate set of goals to command the goal-conditioned agent such that it can collect the most informative data online.

4 Problem Setting: Unlabeled-Offline-to-Online RL

In this paper, we focus on the *unlabeled-offline-to-online RL setting* (Li et al., 2024; Wilcoxson et al., 2025) where the goal is to leverage an unlabeled, reward-free offline dataset collected from $\mathcal{M}$ to learn as sample-efficient as possible online. The offline dataset consists of many trajectory rollouts, $D := \{\tau_1, \tau_2, \dots\}$, where each trajectory rollout τ is a sequence of states and actions $\{s_0, a_0, s_1, a_1, s_2, \dots\}$ that is up to T time steps long. This differs from the standard offline-to-online RL setting (Nakamoto et al., 2024; Zhou et al., 2025) in that the agent has no knowledge of the reward function from the offline dataset. It is also different from the setting considered in Touati et al. (2023) and Kim et al. (2024), where the reward is unknown during offline pre-training but an oracle reward function is accessible either at test-time or during online learning. In our setting, the agent has no knowledge of the task goal or the reward function, and the only way it receives reward information is from interacting with the environment.

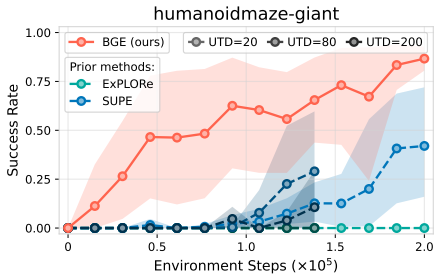


Figure 2: **Prior exploration methods in the unlabeled-offline-to-online RL setting suffer from a sample-efficiency barrier.** SUPE (Wilcoxson et al., 2025) improves when UTD is increased from 20 to 80, but stops improving when UTD is increased from 80 to 200. ExPLORe (Li et al., 2024) completely could not achieve any success. In contrast, our method breaks this barrier and significantly improves the online sample-efficiency.

The sample-efficiency barrier in prior methods. Prior methods (Li et al., 2024; Wilcoxson et al., 2025) tackle the unlabeled offline-to-online RL setting by training a sample-efficient off-policy RL algorithm online. The algorithm performs RL updates on data batch consisting of a mixture of offline data and the online data from the replay buffer. These methods make online exploration and learning efficient by using a high update-to-data (UTD) ratio where the agent takes many (*e.g.*, 20) updates per online environment step. However, there is an empirical limit to how much the UTD can be raised before the training gets unstable. In Figure 2, we report the learning curves of these prior methods when we increase the UTD.

From it, it is clear that there is a limit to how much the prior methods can improve when we increase the UTD ratio. This issue goes deep in how these prior methods adapt to online data. They maintain a single policy that continually chases the optimal behavior under the current reward belief that is constantly changing as the new data comes in. In many cases, the policy needs to first unlearn the previous exploration behavior (*e.g.*, go to the top-left corner of a maze) and then learn the new exploration behavior (*e.g.*, go to the top-right corner of a maze). Such non-stationary learning problems have been previously observed to be challenging for online sample-efficient RL due to plasticity and optimization instability (Nikishin et al., 2022; 2023). Could we somehow *break* such sample-efficiency barrier for online exploration?

5 Batched Goals Exploration

In this section, we describe in detail how our method leverages unsupervised pre-training to break the sample-efficiency barrier by removing the need for online RL updates. At a high-level, our algorithm consists of two key components. **(1) Offline unsupervised pre-training**: we pre-train a goal-conditioned agent and its state occupancy estimator, which allows us to estimate the state

Algorithm 1: Batched Goals Exploration (BGE)

1. Offline Unsupervised Pre-training
for each pretraining step **do**
  Sample batch $B_{\text{offline}} \sim D$;
  Update goal-conditioned agent $\pi_\psi(a \mid s, g)$ on B_{offline};
  Update occupancy estimators $V(g, g^+)$ on B_{offline} using Equations (3) and (4) ;

2. Online Exploration with Adaptive Goal Selection
Initialize $G = \{g_1, g_2, \ldots, g_n\}$ by uniformly sampling n states from D
repeat
  $G \leftarrow \texttt{GoalSelect}(G, D, V)$ (Algorithm 2) ;
  for $g \in G$ **do**
    Rollout $\pi(\cdot \mid s, g)$ to collect n_{traj} trajectories $\{\tau_i\}_{i=1}^{n_{\text{traj}}}$;
    $D_{\text{replay}} \leftarrow D_{\text{replay}} \cup \tau$;
  Update UCB estimate R_{UCB} on all newly collected τ's ;
until $R_{\text{UCB}}(s)$ *converges* $\forall s \in D$;

coverage of the goal-conditioned agent given any commanded goal. **(2) Online exploration with batched goals**: during online exploration, we use the pre-trained occupancy estimator to find a set of goals that maximize the joint state coverage over potential high-reward region. Then, we roll out the goal-conditioned agent conditioned each of these goals to collect more online transitions, which are then used to update our reward belief. Overall, the offline goal-conditioned pre-training decouples the expensive optimization of policy behaviors from high-level goal-directed exploration, allowing our method to perform efficient online exploration without any RL update. We now describe each of these two components in detail (see Algorithm 1 for a pseudocode of our method).

5.1 Offline Unsupervised Pre-training of Goal-conditioned Agent and Occupancy Estimators

We first pre-train a goal-conditioned policy $\pi(a \mid s, g^+)$ on our initial unlabeled offline dataset D to learn goal-reaching capabilities. This allows us to generate a diverse set of behaviors without the need to constantly update the policy like prior methods do (Li et al., 2024; Wilcoxson et al., 2025). To leverage the goal-conditioned agent for effective online data collection, we must select a set of goals from which the agent's chance of discovering novel high-reward regions in the environment is maximized. The novel seeking principle here is akin to that in frontier exploration studied extensively in the purely online goal-conditioned RL literature. However, the novel states in the purely online setting are often hard to reach as the goal-conditioned agent is learning as it explores, resulting in high uncertainty on the expected outcome of goal-directed data collection. Most existing online goal selection techniques resort to simple heuristics (*e.g.*, goal-reaching difficulty (Florensa et al., 2018; Ecoffet et al., 2021), visitation frequency (Pitis et al., 2020), or entropy (Pong et al., 2019)). In contrast, the offline data in our setting contains dynamics information of a diverse set of states (with the only information missing being the reward signal), allowing us to construct accurate estimation of the expected state coverage resulted from rolling out the goal-conditioned agent. By leveraging the offline dataset, we can estimate the expected state coverage in a much more efficient manner.

More specifically, we can pre-train an occupancy value network $V_\phi(s^+, g^+)$ that estimates the probability of visiting state s^+ at any point starting from the initial state distribution ρ when the goal-conditioned agent is conditioned on goal g^+. While directly computing V is challenging, we can learn an intermediate state-action occupancy function $Q_\theta(s, a, s^+, g^+)$ to estimate the probability of visiting state s^+ from state s taking action a when the agent acts according to goal g^+, via the following TD-style objective:

$$L_Q(\theta) = \mathbb{E}_{(s,a,s')\sim D, s^+, g^+ \sim D, a' \sim \pi(\cdot \mid s', g^+)} \left[(Q_\theta(s, a, s^+, g^+) - y)^2 \right], \\ \text{where} \quad y = \mathbb{I}[s = s^+] + (1 - \mathbb{I}[s = s^+])\gamma Q(s', a', s^+, g^+), \quad a' \sim \pi(\cdot \mid s', g^+). \tag{3}$$

The occupancy estimator $V(s^+, g^+)$ can then be estimated from Q with the loss:

$$L_V(\phi) = \mathbb{E}_{s_0 \sim \rho, a_0 \sim \pi(\cdot \mid s_0, g^+)} [(V_\phi(s^+, g^+) - Q(s_0, a_0, s^+, g^+))^2]. \tag{4}$$

Algorithm 2: `GoalSelect`(G, D, V)

Input: Current goal set G, unlabeled offline dataset D, occupancy estimator V.
Hyperparameters: # of optimization steps (N_{opt}), goal proposal size (p), occupancy threshold (σ) ;
for each optimization step **do**
  for each $g^- \in G$ **do**
    Sample a set of new goals: $G^+ = \{g_1^+, \ldots, g_p^+\} \sim D$;
    for each $g^+ \in G^+$ **do**
      $G' \leftarrow \{g^+\} \cup G \setminus g^-$;
      (Propose a new goal set G' by removing g^- and adding g^+)
      $B_{\text{test}} = \{s_1, \cdots, s_m\} \sim D$;
      (Sample a random set of test states)
      Compute the Monte-Carlo estimate of $\text{CoverageScore}(G')$ using B_{test} (Equation (5));
    If any of G' above results in a higher scoring, set $G \leftarrow G'$ for the highest scoring G'
Output: G

After offline pre-training, the occupancy estimator V_ϕ can then be used online to estimate the state coverage that the agent will visit when acting conditioned on each proposed goal in the goal set.

5.2 Online Exploration with Batched Goals

With the pre-trained goal-conditioned agent and its corresponding occupancy estimator, we are now ready to describe our goal selection procedure that optimizes for state coverage over potential high-reward regions. Similar to many curiosity-driven exploration methods, during online exploration, we maintain an upper-confidence bound (UCB) estimate of the reward function, $R_{\text{UCB}}(s) : \mathcal{S} \to \mathbb{R}$, that characterizes whether visiting the state may give the agent high reward. Intuitively, unvisited states exhibit high R_{UCB} values whereas the visited states share the same R_{UCB} as their ground truth reward value in the environment since the uncertainty has been collapsed. The objective of our goal selection procedure is to find a set of commanding goals such that the unique states visited by reaching these goals have a high combined R_{UCB} value:

$$\text{CoverageScore}(G) = \mathbb{E}_{s \sim D}\left[\max_{g^+ \in G} \left[\mathbb{I}\left[V(s, g^+) > \sigma\right]\right] \cdot R_{\text{UCB}}(s)\right], \tag{5}$$

In this objective, we consider a test state s^+ as "covered" if any commanding goal in the goal set G yields an occupancy density $V(s^+, g^+)$ that is sufficiently large (*i.e.*, $V(s^+, g^+) \geq \sigma$). The objective maximizes the sum of all UCB rewards over all "covered" test states. Intuitively, maximizing such objective allows us to explore as many "high-UCB" novel states as possible all at once. This objective not only selects goals adaptively according to the current UCB estimate, but also minimizes overlap between the goal-conditioned occupancies induced by different goals.

To tractably optimize G in Equation (5), we propose a Gibbs-sampling-style procedure that approximates $G^\star = \arg\max_G \text{CoverageScore}(G)$. The procedure uses an iterative optimization process. In each optimization step, we sample a candidate set of goals $G^+ = \{g_1^+, \cdots, g_p^+\} \sim D_{\text{offline}}$ and test whether replacing any goal in G by any goal in G^+ results in a higher $\text{CoverageScore}(G)$ value. Repeating this optimization step allowing us to approximately maximize Equation (5). After the goal optimization procedure, we rollout our goal-conditioned agent for n trajectories conditioned on the selected goals, and update the UCB estimate of the reward function on the new trajectories collected. Altogether, our algorithm leverages the pre-training goal-conditioned agent and occupancy estimator to carefully select a set of goals to make sure the data collected online spreads out over all "high-UCB" novel states without the need for additional policy fine-tuning.

5.3 Practical Implementation Details

Up to now, we have introduced our method and how it leverages offline unsupervised pre-training to enable efficient online exploration without RL updates online. We now describe various implementation details.

UCB rewards. To estimate the upper confidence bound of the reward for each state, $R_{\text{UCB}}(s)$, we take inspiration from the UCB estimation implementation used in Li et al. (2024) and Wilcoxson et al. (2025), where they use a combination of random network distillation (RND) (Burda et al., 2018) reward bonus and the predicted reward from a reward model. We train both the reward model ($R_\xi(s)$) and the RND model ($R_{\text{RND}}(s)$) continually as the online data are collected. The UCB reward can be calculated as follows:

$$R_{\text{UCB}}(s) := R_\xi(s) + \beta R_{\text{RND}}(s), \tag{6}$$

where β is a tunable hyperparameter that controls the strength of the RND exploration bonus.

Goal representation. Following prior work, we assume access to an oracle goal representation $\Psi : \mathcal{S} \to \mathcal{G}$ (Park et al., 2025b), and that the reward function R takes the form of $R(s, a) = R_g(\Psi(s))$ for some function $R_g : \mathcal{G} \to \mathbb{R}$. This allows us to simplify the implementation of UCB reward R_{UCB}, occupancy estimators Q and V. For R_{UCB}, we use $\Psi(s)$ instead of s, and maintain the UCB estimate directly in the goal space: $R_{\text{UCB}}(\Psi(s))$. For Q and V, we use $\Psi(s^+)$ instead of s^+ and learn $Q(s, a, \Psi(s^+), g^+), V(\Psi(s^+), g^+)$ directly. To evaluate our pre-trained agent, we rollout our pre-trained goal-conditioned agent by commanding the goal g_{eval} that corresponds to the reward-maximizing state according to our reward model: $g_{\text{eval}} = \Psi(\arg\max_{s \in D} R_{\text{RM}}(s))$.

Adaptive σ and test state filtering. Empirically, it is often helpful to use an adaptive threshold for determining whether a state is being "covered". In particular, we find that setting σ to $V(g^+, g^+)$ often removes the need for tuning σ. In addition, to allow our exploration goals to focus on the high-reward states more, we perform a test state filtering where only sample test states that have estimated rewards higher than the highest reward seen in the online replay buffer:

$$\tilde{D} \leftarrow \left\{ s \mid s \in D : R_\xi(s) > \max_{s_{\text{replay}} \in D_{\text{replay}}} R(s_{\text{replay}}) - \varepsilon \right\}, \tag{7}$$

where ε is set to 0.6 in our experiments. With this formulation, we find that we can also replace the $R_{\text{UCB}}(s)$ in Equation (5) with $R_{\text{RND}}(s)$ such that we no longer need to tune β. The practical scoring function we use in our experiments is

$$\text{PracticalCoverageScore}(G) = \mathbb{E}_{s \sim \tilde{D}} \left[\max_{g^+ \in G} \left[\mathbb{I}\left[V(\Psi(s), g^+) > V(g^+, g^+) \right] \right] \cdot R_{\text{RND}}(s) \right]. \tag{8}$$

See additional implementation details in Appendix C.

6 Experimental Setup

Our experiments are designed to evaluate how effective our method is in the *unlabeled-offline-to-online RL setting* (described in Section 4). We focus on sparse-reward domains with a large state space and compare our method with prior methods.

6.1 Domains and Datasets

We consider 4 domains from OGBench (Park et al., 2025a) with varying levels of difficulty: `antmaze-large`, `antmaze-giant`, `humanoidmaze-giant`, `cube-double`. For all locomotion domains (*i.e.*, `ant` or `humanoid`), we use the default `navigate`-style dataset. For the `cube` domain, we use the default `play`-style dataset. These tasks requires long-horizon reasoning to solve. For example, the hardest task in `humanoidmaze-giant` requires up to 3000 environment steps to solve, and the agent only receives a positive reward signal on reaching the goal location (that is unknown to the agent). For these tasks, informative and adaptive exploration behaviors are critical for the agent to find the goal in a sample-efficient manner, making them ideal for stress-testing exploration algorithms. For each domain, we use the default task (`task-2` for `cube` and `task-1` for the rest).

See additional domain details and visualizations in Appendix B.

6.2 Comparisons

For our first set of comparisons, we first consider the previous state-of-the-art methods in the *unlabeled-offline-to-online RL* setting:

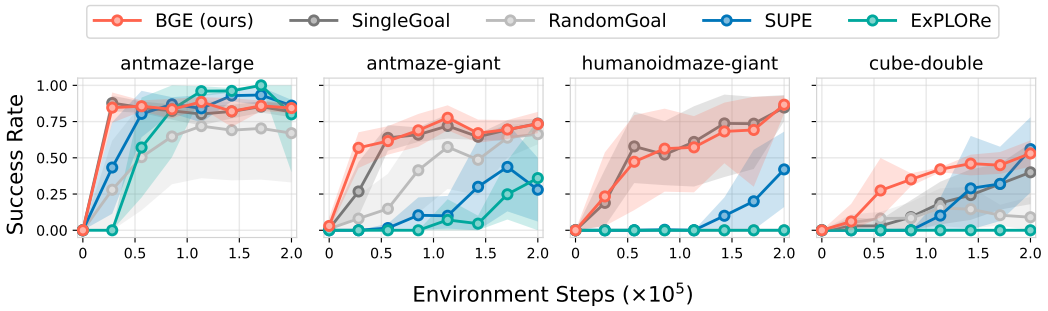


Figure 3: **Main results (5 seeds with 95% CI).** BGE achieves better online sample-efficiency compared to prior best exploration methods in the unlabeled-offline-to-online RL setting.

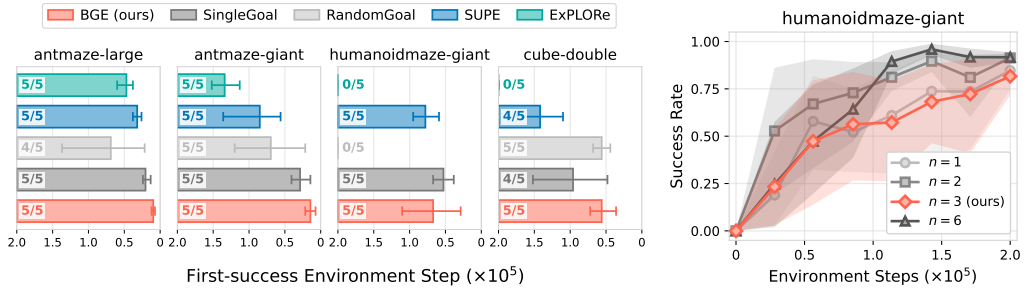


Figure 4: **First-success analysis and sensitivity analysis.** *left*: *Ours* achieves the first success faster than prior methods. Aside from the first success environment step, we also report the number of seeds (out of 5) that the method achieves at least one success during online exploration in 200K environment steps. *right*: Increasing n to 6 does not further improve the performance of our algorithm.

ExPLORe (Li et al., 2024) run online soft-actor critic (SAC) (Haarnoja et al., 2018) on both offline and online replay buffer with high UTD akin to RLPD (Ball et al., 2023). This baseline maintains a UCB reward model similar to our method and uses it to assign reward label in the offline training batch on the fly.

SUPE (Wilcoxson et al., 2025) improves upon the first baseline by incorporating offline unsupervised skill pre-training and running the same online RL algorithm on a high-level skill space to generate more temporally coherent exploration behaviors.

Both of them rely on frequent updates of an online off-policy RL agent to adapt to a changing reward belief (as modeled by the UCB reward model), which we show in Figure 2 to suffer from a sample-efficiency barrier. Aside from these prior methods, we also consider several ablation baselines where the same goal-conditioned pre-training is used and *no RL updates* are used.

RandomGoal is the same as our algorithm except we rollout the goal-conditioned agent with goals randomly sampled from the offline data. This tests whether selecting goals using the occupancy estimator results in better online exploration.

SingleGoal is the same as our algorithm except we only optimize for a goal set of size 1 ($n = 1$). This baseline tests whether having a batch of goals to explore simultaneously yields better exploration.

7 Results

We present our empirical results in this section organized by the following questions:

Q1. Does our method lead to better sample-efficiency compared to prior unlabeled offline to online methods?

Figure 3 reports the online sample-efficiency of our method compared to prior methods across 4 domains. Compared to prior SOTA methods, SUPE and ExPLORe, BGE consistently ob-

tains better sample-efficiency. To provide more insights on why BGE explores well, we also report how long it takes for the agent to obtain the first success in the environment online in Figure 4. Our method generally achieves the first success earlier than baselines, with the exception on `humanoidmaze-giant` where our method achieves the first success similar to SUPE and slightly slower than SingleGoal, and on `cube-double` where our method achieves the first success similar to the RandomGoal baseline. This suggests that our method explores more effectively than prior methods and our ablation baselines, which leads to better sample-efficiency. Overall, our method is the most consistent, and the only method that always succeeds ($5/5$ seeds on all domains).

Q2. How sensitive is our algorithm with respective to the goal set size? We analyze the sensitivity of our algorithm with respect to its hyperparameters. Figure 4 (right) reports the performance of our algorithm when we vary the size of the goal set on `humanoidmaze-giant`. While we observe some variability across individual runs, our method is largely robust to the goal set size.

Q3. How computational efficient and deployment efficient is our algorithm?

Our algorithm is designed to work without relying on online RL updates, and naturally it only incurs a fraction of the computational cost compared to prior methods that rely on online RL updates (as shown in Table 1, first column). Another side benefit of our algorithm is that we do not make frequent changes to the agent's online exploration behavior (the behavior only changes when the commanding goals change). This is often a desirable property in real-world settings when each change involves a big deployment cost (*e.g.*, iterated offline RL/batch online RL in robotics (Dong et al., 2025; Intelligence et al., 2025)). Our algorithm is extremely deployment efficient (as shown in Table 1, second column), making it promising to be adopted in these realistic settings.

Method	Online Training Clock Time	# of Deployments
ExPLORe (Li et al., 2024)	$\approx$ 2 hours	2×10^5 (every env step)
SUPE (Wilcoxson et al., 2025)	$\approx$ 7 hours	2×10^5 (every env step)
BGE (ours)	$\approx$ 80 minutes	9

Table 1: **Training time and # of deployments for each method on `humanoidmaze-giant`.**

8 Discussion

We propose a new online exploration algorithm, BGE, that leverages offline goal-conditioned pre-training to break the sample-efficiency barrier in prior methods in the unlabeled-offline-to-online RL setting. Several challenges remain, suggesting promising directions for future research. For example, our method uses a frozen, pre-trained goal-conditioned agent, which may struggle on tasks that require drastically different exploration behaviors obtainable from the prior data. It is foreseeable that developing a good recipe for sample-efficient online fine-tuning of goal-conditioned agent could further boost the sample-efficiency of our algorithm. See more discussions and limitations in Appendix F.

AI Use Statement

In this work, we used generative AI tools for assisting with implementing methods. We have not used generative AI tools for generating synthetic data sets, helping develop theoretical models or conceptual frameworks, formulating mathematical claims, providing critical ingredients for proving mathematical claims, assisting in the writing of proofs, proposing hypotheses, designing or providing feedback on research methodology or experiments, assisting with translation, cleaning and reformat dataset, supporting qualitative and thematic data analysis, and interpreting results. Additionally, we used generative AI tools for creating and modifying scientific figures and images. We have reviewed all AI-assisted work. LLM-generated code was verified and tested for correctness, and all scientific figures and images were checked for accuracy. We take responsibility for the final content of this work, including text, claims or artifacts produced with the aid of generative AI.

Reproducibility statement

To ensure reproducibility of our results, Section 5 describes the methdo in full, with pseudocode in Algorithm 1 and Algorithm 2, and the pre-training objectives in Equation (3) and **??**. Practical implementation choices, including UCB reward, goal representation, and adaptive thresholding are outlined in Section 5 and further detailed in Section 5.3. All hyperparameters for goal-conditioned pre-training for each method and domain are listed in Appendix E.

References

Joshua Achiam and Shankar Sastry. Surprise-based intrinsic motivation for deep reinforcement learning. *arXiv preprint arXiv:1703.01732*, 2017.

Rishabh Agarwal, Max Schwarzer, Pablo Samuel Castro, Aaron Courville, and Marc G. Bellemare. Reincarnating reinforcement learning: Reusing prior computation to accelerate progress, 2022. URL https://arxiv.org/abs/2206.01626.

Marcin Andrychowicz, Filip Wolski, Alex Ray, Jonas Schneider, Rachel Fong, Peter Welinder, Bob McGrew, Josh Tobin, Pieter Abbeel, and Wojciech Zaremba. Hindsight experience replay, 2018. URL https://arxiv.org/abs/1707.01495.

Junik Bae, Kwanyoung Park, and Youngwoon Lee. Tldr: Unsupervised goal-conditioned rl via temporal distance-aware representations. *arXiv preprint arXiv:2407.08464*, 2024.

Philip J. Ball, Laura Smith, Ilya Kostrikov, and Sergey Levine. Efficient online reinforcement learning with offline data, 2023. URL https://arxiv.org/abs/2302.02948.

Marc G. Bellemare, Sriram Srinivasan, Georg Ostrovski, Tom Schaul, David Saxton, and Remi Munos. Unifying count-based exploration and intrinsic motivation, 2016. URL https://arxiv.org/abs/1606.01868.

Yuri Burda, Harrison Edwards, Amos Storkey, and Oleg Klimov. Exploration by random network distillation, 2018. URL https://arxiv.org/abs/1810.12894.

Elliot Chane-Sane, Cordelia Schmid, and Ivan Laptev. Goal-conditioned reinforcement learning with imagined subgoals. In Marina Meila and Tong Zhang (eds.), *Proceedings of the 38th International Conference on Machine Learning*, volume 139 of *Proceedings of Machine Learning Research*, pp. 1430–1440. PMLR, 18–24 Jul 2021. URL https://proceedings.mlr.press/v139/chane-sane21a.html.

Yevgen Chebotar, Karol Hausman, Yao Lu, Ted Xiao, Dmitry Kalashnikov, Jake Varley, Alex Irpan, Benjamin Eysenbach, Ryan Julian, Chelsea Finn, et al. Actionable models: Unsupervised offline reinforcement learning of robotic skills. *arXiv preprint arXiv:2104.07749*, 2021.

Jongwook Choi, Archit Sharma, Honglak Lee, Sergey Levine, and Shixiang Shane Gu. Variational empowerment as representation learning for goal-based reinforcement learning. *arXiv preprint arXiv:2106.01404*, 2021.

Perry Dong, Suvir Mirchandani, Dorsa Sadigh, and Chelsea Finn. What matters for batch online reinforcement learning in robotics? *arXiv preprint arXiv:2505.08078*, 2025.

Ron Dorfman, Idan Shenfeld, and Aviv Tamar. Offline meta reinforcement learning–identifiability challenges and effective data collection strategies. *Advances in Neural Information Processing Systems*, 34:4607–4618, 2021.

Pierluca D'Oro, Max Schwarzer, Evgenii Nikishin, Pierre-Luc Bacon, Marc G Bellemare, and Aaron Courville. Sample-efficient reinforcement learning by breaking the replay ratio barrier. In *The Eleventh International Conference on Learning Representations*, 2023. URL https://openreview.net/forum?id=OpC-9aBBVJe.

Yan Duan, John Schulman, Xi Chen, Peter L Bartlett, Ilya Sutskever, and Pieter Abbeel. RL^2: Fast reinforcement learning via slow reinforcement learning. *arXiv preprint arXiv:1611.02779*, 2016.

Adrien Ecoffet, Joost Huizinga, Joel Lehman, Kenneth O. Stanley, and Jeff Clune. Go-explore: a new approach for hard-exploration problems, 2021. URL https://arxiv.org/abs/1901.10995.

Aleksandr Ermolov and Nicu Sebe. Latent world models for intrinsically motivated exploration. *Advances in Neural Information Processing Systems*, 33:5565–5575, 2020.

Benjamin Eysenbach, Abhishek Gupta, Julian Ibarz, and Sergey Levine. Diversity is all you need: Learning skills without a reward function. In *International Conference on Learning Representations*, 2019. URL https://openreview.net/forum?id=SJx63jRqFm.

Carlos Florensa, David Held, Xinyang Geng, and Pieter Abbeel. Automatic goal generation for reinforcement learning agents. In *International conference on machine learning*, pp. 1515–1528. PMLR, 2018.

Sébastien Forestier, Rémy Portelas, Yoan Mollard, and Pierre-Yves Oudeyer. Intrinsically motivated goal exploration processes with automatic curriculum learning, 2022. URL https://arxiv.org/abs/1708.02190.

Tuomas Haarnoja, Aurick Zhou, Pieter Abbeel, and Sergey Levine. Soft actor-critic: Off-policy maximum entropy deep reinforcement learning with a stochastic actor. In *International conference on machine learning*, pp. 1861–1870. PMLR, 2018.

David Held, Xinyang Geng, Carlos Florensa, and Pieter Abbeel. Automatic goal generation for reinforcement learning agents, 2018. URL https://openreview.net/forum?id=SyhRVm-Rb.

Rein Houthooft, Xi Chen, Yan Duan, John Schulman, Filip De Turck, and Pieter Abbeel. Vime: Variational information maximizing exploration, 2017. URL https://arxiv.org/abs/1605.09674.

Hao Hu, Yiqin Yang, Jianing Ye, Ziqing Mai, and Chongjie Zhang. Unsupervised behavior extraction via random intent priors. In *Thirty-seventh Conference on Neural Information Processing Systems*, 2023. URL https://openreview.net/forum?id=4vGVQVz5KG.

Marcel Hussing, Claas Voelcker, Igor Gilitschenski, Amir massoud Farahmand, and Eric Eaton. Dissecting deep rl with high update ratios: Combatting value divergence, 2024. URL https://arxiv.org/abs/2403.05996.

Physical Intelligence, Ali Amin, Raichelle Aniceto, Ashwin Balakrishna, Kevin Black, Ken Conley, Grace Connors, James Darpinian, Karan Dhabalia, Jared DiCarlo, et al. $\pi^{\star}_{0.6}$: a vla that learns from experience. *arXiv preprint arXiv:2511.14759*, 2025.

Whiyoung Jung, Giseung Park, and Youngchul Sung. Population-guided parallel policy search for reinforcement learning, 2020. URL https://arxiv.org/abs/2001.02907.

Leslie Pack Kaelbling. Learning to achieve goals. In *IJCAI*, volume 2, pp. 1094–8, 1993.

Junsu Kim, Seohong Park, and Sergey Levine. Unsupervised-to-online reinforcement learning, 2024. URL https://arxiv.org/abs/2408.14785.

Michael Laskin, Denis Yarats, Hao Liu, Kimin Lee, Albert Zhan, Kevin Lu, Catherine Cang, Lerrel Pinto, and Pieter Abbeel. Urlb: Unsupervised reinforcement learning benchmark, 2021. URL https://arxiv.org/abs/2110.15191.

Seunghyun Lee, Younggyo Seo, Kimin Lee, Pieter Abbeel, and Jinwoo Shin. Offline-to-online reinforcement learning via balanced replay and pessimistic q-ensemble, 2021. URL https://arxiv.org/abs/2107.00591.

Sergey Levine, Aviral Kumar, George Tucker, and Justin Fu. Offline reinforcement learning: Tutorial, review, and perspectives on open problems. *arXiv preprint arXiv:2005.01643*, 2020.

Qiyang Li, Jason Zhang, Dibya Ghosh, Amy Zhang, and Sergey Levine. Accelerating exploration with unlabeled prior data. *Advances in Neural Information Processing Systems*, 36, 2024.

Qiyang Li, Zhiyuan Zhou, and Sergey Levine. Reinforcement learning with action chunking. *arXiv preprint arXiv:2507.07969*, 2025.

Qiyang Li, Seohong Park, and Sergey Levine. Decoupled q-chunking. In *The Fourteenth International Conference on Learning Representations*, 2026. URL https://openreview.net/forum?id=aqGNdZQL9l.

Hao Liu and Pieter Abbeel. Behavior from the void: Unsupervised active pre-training. In A. Beygelzimer, Y. Dauphin, P. Liang, and J. Wortman Vaughan (eds.), *Advances in Neural Information Processing Systems*, 2021. URL https://openreview.net/forum?id=fIn4wLS2XzU.

Minghuan Liu, Menghui Zhu, and Weinan Zhang. Goal-conditioned reinforcement learning: Problems and solutions, 2022. URL https://arxiv.org/abs/2201.08299.

Sam Lobel, Akhil Bagaria, and George Konidaris. Flipping coins to estimate pseudocounts for exploration in reinforcement learning, 2023. URL https://arxiv.org/abs/2306.03186.

Russell Mendonca, Oleh Rybkin, Kostas Daniilidis, Danijar Hafner, and Deepak Pathak. Discovering and achieving goals via world models. *Advances in neural information processing systems*, 34:24379–24391, 2021.

Josh Merel, Leonard Hasenclever, Alexandre Galashov, Arun Ahuja, Vu Pham, Greg Wayne, Yee Whye Teh, and Nicolas Heess. Neural probabilistic motor primitives for humanoid control. *arXiv preprint arXiv:1811.11711*, 2018.

Eric Mitchell, Rafael Rafailov, Xue Bin Peng, Sergey Levine, and Chelsea Finn. Offline meta-reinforcement learning with advantage weighting. In *International Conference on Machine Learning*, pp. 7780–7791. PMLR, 2021.

Ashvin Nair, Vitchyr Pong, Murtaza Dalal, Shikhar Bahl, Steven Lin, and Sergey Levine. Visual reinforcement learning with imagined goals, 2018. URL https://arxiv.org/abs/1807.04742.

Mitsuhiko Nakamoto, Yuexiang Zhai, Anikait Singh, Max Sobol Mark, Yi Ma, Chelsea Finn, Aviral Kumar, and Sergey Levine. Cal-ql: Calibrated offline rl pre-training for efficient online fine-tuning, 2024. URL https://arxiv.org/abs/2303.05479.

Evgenii Nikishin, Max Schwarzer, Pierluca D'Oro, Pierre-Luc Bacon, and Aaron Courville. The primacy bias in deep reinforcement learning. In *International conference on machine learning*, pp. 16828–16847. PMLR, 2022.

Evgenii Nikishin, Junhyuk Oh, Georg Ostrovski, Clare Lyle, Razvan Pascanu, Will Dabney, and André Barreto. Deep reinforcement learning with plasticity injection. *Advances in Neural Information Processing Systems*, 36:37142–37159, 2023.

Georg Ostrovski, Marc G Bellemare, Aäron Oord, and Rémi Munos. Count-based exploration with neural density models. In *International conference on machine learning*, pp. 2721–2730. PMLR, 2017.

Vincenzo De Paola, Riccardo Zamboni, Mirco Mutti, and Marcello Restelli. Enhancing diversity in parallel agents: A maximum state entropy exploration story, 2025. URL https://arxiv.org/abs/2505.01336.

Seohong Park, Dibya Ghosh, Benjamin Eysenbach, and Sergey Levine. HIQL: Offline goal-conditioned RL with latent states as actions. In *Thirty-seventh Conference on Neural Information Processing Systems*, 2023. URL https://openreview.net/forum?id=cLQCCtVDuW.

Seohong Park, Kevin Frans, Benjamin Eysenbach, and Sergey Levine. OGBench: Benchmarking offline goal-conditioned RL. In *The Thirteenth International Conference on Learning Representations*, 2025a. URL https://openreview.net/forum?id=M992mjgKzI.

Seohong Park, Kevin Frans, Deepinder Mann, Benjamin Eysenbach, Aviral Kumar, and Sergey Levine. Horizon reduction makes RL scalable. In *The Thirty-ninth Annual Conference on Neural Information Processing Systems*, 2025b. URL https://openreview.net/forum?id=hguaupzLCU.

Deepak Pathak, Pulkit Agrawal, Alexei A. Efros, and Trevor Darrell. Curiosity-driven exploration by self-supervised prediction, 2017. URL https://arxiv.org/abs/1705.05363.

Silviu Pitis, Harris Chan, Stephen Zhao, Bradly Stadie, and Jimmy Ba. Maximum entropy gain exploration for long horizon multi-goal reinforcement learning, 2020. URL https://arxiv.org/abs/2007.02832.

Vitchyr H Pong, Murtaza Dalal, Steven Lin, Ashvin Nair, Shikhar Bahl, and Sergey Levine. Skew-fit: State-covering self-supervised reinforcement learning. *arXiv preprint arXiv:1903.03698*, 2019.

Vitchyr H. Pong, Murtaza Dalal, Steven Lin, Ashvin Nair, Shikhar Bahl, and Sergey Levine. Skew-fit: State-covering self-supervised reinforcement learning, 2020. URL https://arxiv.org/abs/1903.03698.

Vitchyr H Pong, Ashvin V Nair, Laura M Smith, Catherine Huang, and Sergey Levine. Offline meta-reinforcement learning with online self-supervision. In *International Conference on Machine Learning*, pp. 17811–17829. Pmlr, 2022.

Rafael Rafailov, Kyle Hatch, Victor Kolev, John D. Martin, Mariano Phielipp, and Chelsea Finn. Moto: Offline pre-training to online fine-tuning for model-based robot learning, 2024. URL https://arxiv.org/abs/2401.03306.

Kate Rakelly, Aurick Zhou, Chelsea Finn, Sergey Levine, and Deirdre Quillen. Efficient off-policy meta-reinforcement learning via probabilistic context variables. In *International conference on machine learning*, pp. 5331–5340. PMLR, 2019.

Tom Schaul, Daniel Horgan, Karol Gregor, and David Silver. Universal value function approximators. In Francis Bach and David Blei (eds.), *Proceedings of the 32nd International Conference on Machine Learning*, volume 37 of *Proceedings of Machine Learning Research*, pp. 1312–1320, Lille, France, 07–09 Jul 2015a. PMLR. URL https://proceedings.mlr.press/v37/schaul15.html.

Tom Schaul, Daniel Horgan, Karol Gregor, and David Silver. Universal value function approximators. In Francis Bach and David Blei (eds.), *Proceedings of the 32nd International Conference on Machine Learning*, volume 37 of *Proceedings of Machine Learning Research*, pp. 1312–1320, Lille, France, 07–09 Jul 2015b. PMLR. URL https://proceedings.mlr.press/v37/schaul15.html.

Archit Sharma, Shixiang Gu, Sergey Levine, Vikash Kumar, and Karol Hausman. Dynamics-aware unsupervised discovery of skills. In *International Conference on Learning Representations*, 2020. URL https://openreview.net/forum?id=HJgLZR4KvH.

Ghada Sokar, Rishabh Agarwal, Pablo Samuel Castro, and Utku Evci. The dormant neuron phenomenon in deep reinforcement learning. In *International Conference on Machine Learning*, pp. 32145–32168. PMLR, 2023.

Yuda Song, Yifei Zhou, Ayush Sekhari, Drew Bagnell, Akshay Krishnamurthy, and Wen Sun. Hybrid RL: Using both offline and online data can make RL efficient. In *The Eleventh International Conference on Learning Representations*, 2023. URL https://openreview.net/forum?id=yyBis80iUuU.

Bradly C. Stadie, Sergey Levine, and Pieter Abbeel. Incentivizing exploration in reinforcement learning with deep predictive models, 2015. URL https://arxiv.org/abs/1507.00814.

Haoran Tang, Rein Houthooft, Davis Foote, Adam Stooke, OpenAI Xi Chen, Yan Duan, John Schulman, Filip DeTurck, and Pieter Abbeel. # exploration: A study of count-based exploration for deep reinforcement learning. *Advances in neural information processing systems*, 30, 2017.

Ahmed Touati, Jérémy Rapin, and Yann Ollivier. Does zero-shot reinforcement learning exist? In *The Eleventh International Conference on Learning Representations*, 2023. URL https://openreview.net/forum?id=MYEap_OcQI.

Andrew Wagenmaker, Zhiyuan Zhou, and Sergey Levine. Behavioral exploration: Learning to explore via in-context adaptation. In *Forty-second International Conference on Machine Learning*, 2025.

Max Wilcoxson, Qiyang Li, Kevin Frans, and Sergey Levine. Leveraging skills from unlabeled prior data for efficient online exploration. In *International Conference on Machine Learning (ICML)*, 2025.

Maciej Wołczyk, Bartłomiej Cupiał, Mateusz Ostaszewski, Michał Bortkiewicz, Michał Zając, Razvan Pascanu, Łukasz Kuciński, and Piotr Miłoś. Fine-tuning reinforcement learning models is secretly a forgetting mitigation problem, 2024. URL https://arxiv.org/abs/2402.02868.

Tengyang Xie, Nan Jiang, Huan Wang, Caiming Xiong, and Yu Bai. Policy finetuning: Bridging sample-efficient offline and online reinforcement learning, 2022. URL https://arxiv.org/abs/2106.04895.

Haichao Zhang, Wei Xu, and Haonan Yu. Policy expansion for bridging offline-to-online reinforcement learning. In *The Eleventh International Conference on Learning Representations*, 2023. URL https://openreview.net/forum?id=-Y34L45JR6z.

Han Zheng, Xufang Luo, Pengfei Wei, Xuan Song, Dongsheng Li, and Jing Jiang. Adaptive policy learning for offline-to-online reinforcement learning. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 37, pp. 11372–11380, 2023.

Zhiyuan Zhou, Andy Peng, Qiyang Li, Sergey Levine, and Aviral Kumar. Efficient online reinforcement learning fine-tuning need not retain offline data, 2025. URL https://arxiv.org/abs/2412.07762.

Luisa Zintgraf, Kyriacos Shiarlis, Maximilian Igl, Sebastian Schulze, Yarin Gal, Katja Hofmann, and Shimon Whiteson. Varibad: A very good method for bayes-adaptive deep rl via meta-learning. *arXiv preprint arXiv:1910.08348*, 2019.

A Compute Resource

We run all our experiments on NVIDIA A5000 GPUs. The results in this paper require approximately 600 GPU hours to run.

B Domain Detail

B.1 OGBench antmaze

Antmaze is a navigation task that involves controlling an ant with 8 degrees of freedom through a maze. It is a standard maze task that involves an ant traveling around the maze to reach a predetermined goal, with a reward of 0 at the goal and rewards of -1 everywhere else. We run our method on two of the mazes: `antmaze-large` and `antmaze-giant`. We use the `navigate` dataset, which consists of trajectories of the ant traveling around the maze.

B.2 OGBench humanoidmaze

Humanoidmaze is a long-horizon navigation task similar to that of `antmaze`, with a reward of 0 at the goal of the maze and rewards of -1 everywhere else. In contrast to `antmaze`, `humanoidmaze` involves controlling 21 degrees of freedom rather than 8. We run our method on `humanoidmaze-giant`, which can require up to 3000 steps.

B.3 OGBench cube

Cube is a manipulation task that involves picking and placing unique blocks in specified locations. We run our method on `cube-double`, which involves moving and stacking two cubes. The goal is a specific cube configuration. To succeed, the agent must place both of the cubes in their correct locations. When none of the cubes are placed in their correct locations, the agent receives a reward of -2. When one of the cubes are in their correct locations, the agent receives a reward of -1. When both of the cubes are correctly placed, the agent receives a reward of 0.

C Implementation Details

In this section, we describe implementation details of our method and baselines. See the hyperparmeters in Table 3.

C.1 BGE

Our goal-conditioned pre-training hyperparameters are available in Table 2.

In equation 3, we check for exact equality in $s = s^\star$, which is possible even in continuous spaces due to how we construct input pairs for training. During pre-training, to provide inputs to train the goal-conditioned agent and occupancy networks, we match states from the offline dataset together to form goal-state pairs. Then, $\mathbb{I}[s = s^\star]$ is 1 only when in the input, a state is matched to itself. This is a standard relabeling trick used in offline goal-conditioned RL papers (*Parket al.*, 2025*a*).

To encourage our evaluation goals to be in the center of the high reward states, rather than directly calculating the estimated reward of each proposed evaluation goal using the reward network, we add noise to each proposed goal, estimating the mean of the reward of the goal and the reward of the goal plus random noise.

To train effective occupancy networks, we find that we need a larger discount rate to account for the probability of future states visited than used to train the agent. The discount rate we use is listed in Table 3.

For `cube-double`, we often find the occupancy networks are sensitive to σ, and sometimes using the heuristics $\sigma = V(g^+, g^+)$ results in the failure mode where too many states are considered being "covered". To address this issue, whenever we find more than 1% of the states are considered to be "covered", we only consider the top 1% of the states (s) ranked by their $V(s, g^+)$ to be covered

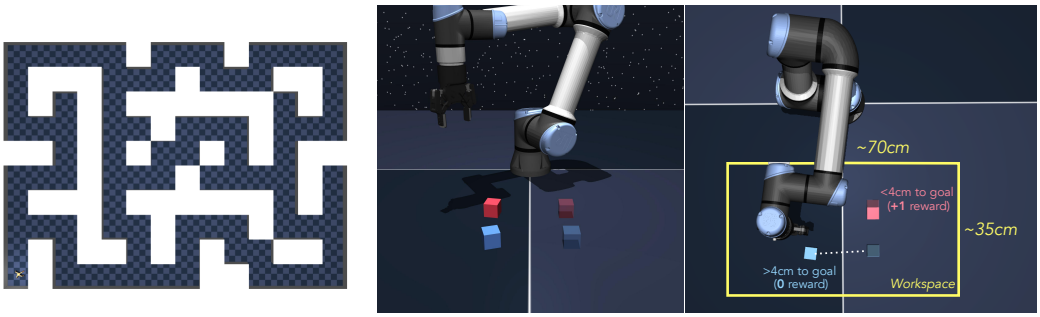


Figure 5: **We evaluate our method on long-horizon, sparse reward task domains from OGBench (Park et al., 2025a).** In `antmaze-giant` and `humanoidmaze-giant` (*left*), the RL agent is tasked to control an ant/humanoid to navigate in a large, complex maze to an unknown goal location. Solving the hardest task can take up to 2000 and 3000 environment step for ant and humanoid respectively. In `cube-double` domain, the RL agent is tasked to manipulate colored blocks to some unknown configurations (*middle*). The desired cube location can be anywhere within the 35cm $\times$ 70cm workspace and the environment gives +1 reward for every cube that is within 4cm distance of its desired location (*right*). The task is only considered successful when all cubes have been moved to their desired locations.

instead. We also find it helpful to augment the sum of the estimated reward of all proposed goals, $\sum_{g^+ \sim G} [R_\xi(g^+)]$, to PracticalCoverageScore scoring the goal sets.

C.2 ExPLORe

ExPLORe (Li et al., 2024) runs RLPD where it leverages unlabeled offline data by optimistically labeling the offline data using a UCB reward model. The UCB reward model consists of a standard reward model (R_ξ) augmented by an RND model (R_{RND}). The UCB reward is defined the same way as our method:

$$R_{\text{UCB}}(s) = R_\xi(s) + \beta \text{R}_{\text{RND}}(s). \tag{9}$$

We use the same set of hyperparameters used in SUPE (Wilcoxson et al., 2025) for this baseline.

C.3 SUPE

SUPE (Wilcoxson et al., 2025) is similar to ExPLORe, but first pre-trains a space of skill policy on the unlabeled prior data. Then SUPE runs RLPD directly on the skill latent action space instead of the raw action space. We use the official implementation and its hyperparameters in all our experiments.

D Additional Related Work

Unsupervised reinforcement learning aims to pre-train agents using reward-free interactions to acquire diverse behaviors or representations that accelerate downstream learning, often using self-supervised intrinsic rewards (Laskin et al., 2021). These methods pursue various objectives, from latent skill discovery (Eysenbach et al., 2019; Sharma et al., 2020) to state coverage via entropy maximization (Liu & Abbeel, 2021). One popular choice of pre-training skill is goal-conditioned skills, where the policy, when given a goal state, is pre-trained to maximize its probability to reach such state. Within this framework, goal proposal techniques (Forestier et al., 2022; Andrychowicz et al., 2018; Held et al., 2018; Pitis et al., 2020; Pong et al., 2020; Chane-Sane et al., 2021; Ecoffet et al., 2021) attempt to generate a curriculum of useful goals for a goal-conditioned agent to learn. Many prior works use goal proposal to aid exploration to prepare the agent to solve future tasks, encouraging it to learn diverse skills by randomly sampling goals in a learned latent space (Nair et al., 2018; Pong et al., 2020) or by maximizing the entropy of the goal distribution to ensure broad state coverage (Pitis et al., 2020). While most of these existing goal-proposal methods focus on the purely online setting, we instead focus on the setting where we have access to an offline dataset. Having access to such prior information allows our method to pre-train an accurate occupancy estimator for

the goal-conditioned agent, such that our proposed goals can better guide the agent to reach novel states.

Meta reinforcement learning methods also enable training-free online exploration. These methods (Duan et al., 2016; Zintgraf et al., 2019) often assume access to a prior task distribution from which they meta-learn an adaptive policy that conditions on the online data collected so far, and produces appropriate exploration behaviors. This idea has also been brought to the offline setting where the policy can be learned completely offline on some prior data (Rakelly et al., 2019; Dorfman et al., 2021; Mitchell et al., 2021; Pong et al., 2022; Wagenmaker et al., 2025). While these methods hold the promise of providing an approximated Bayes-optimal exploration policy, they are often much harder to optimize and require a well-specified prior or meta-training distribution. In contrast, our method does not rely on training a policy to be adaptive on a prior task distribution. Instead, our algorithm relies on an online batched goals optimization procedure to composes a set of pre-trained goal-directed behaviors that jointly maximizes coverage of novel states directly.

E HYPERPARAMETERS

Parameter	Value
Batch size	256 except 4096 for `humanoidmaze-giant`
Discount factor (γ)	0.99 for `cube/antmaze-large` 0.995 for `antmaze-giant` and 0.999 for `humanoidmaze-giant`
Pretraining agent	FQL; except DQC (Li et al., 2026) for `{humanoid}-*`
Action Chunking	No; except `cube` where action chunk size of 5 is used
Best-of-N sampling	4; except 32 for `humanoid`
Optimizer	Adam
Learning rate	3×10^{-4}
Target network update rate (λ)	5×10^{-3}
Network width	512; except 1024 for `humanoidmaze-giant, cube-double`
Network depth	4 hidden layers

Table 2: **Goal-conditioned pre-training hyperparameters.** For the DQC agent, we directly use the same hyperparameter as used in Li et al. (2026). For the FQL agent, we the implementation from Park et al. (2025b) with the $\alpha = 300$ for `cube`, and $\alpha = 10$ for other domains, and additionally use best-of-N sampling (similar to BFN from Li et al. (2025)) with a $N = 4$.

Domains	ExPLORe (β, UTD)	SUPE $(\beta, \beta_{\text{KL}}, \text{UTD})$	BGE (n, n_{traj})
`antmaze-large-*`	$(2, 20)$	$(2, 0.1, 20)$	$(3, 2)$
`antmaze-giant-*`	$(2, 20)$	$(2, 0.1, 20)$	$(3, 2)$
`humanoidmaze-giant-*`	$(2, 20)$	$(2, 0.1, 20)$	$(3, 2)$
`cube-single-*`	$(2, 20)$	$(2, 0.2, 20)$	$(10, 2)$
`cube-double-*`	$(2, 20)$	$(2, 0.2, 20)$	$(10, 4)$

Table 3: **Domain-specific hyperparameters.** We use the same set of hyperparameters for all five tasks in each domain. β controls the strength of RND in the UCB reward model for both ExPLORe and SUPE. β_{KL} controls the KL regularization strength for the VAE skill training in SUPE.

F LIMITATIONS

Our method is dependent on offline data coverage. That is, both the candidate goals and the occupancy estimates in our method come from the offline dataset. If the data rarely visits where the reward lies, the goal-conditioned agent cannot reliably reach the reward and the occupancy estimator has little training signal there. Furthermore, all of BGE's online adaptivity comes from goal selection, not from policy improvement. As a result, BGE can find the reward-maximizing region

only if the pre-trained agent can reach it, and its final performance is bounded by how well the agent reaches the correct goal. Finally, the pre-trained agent is trained to reach goals effectively, not to maximize state coverage. Thus, a near-optimal goal-reaching policy would take the shortest path to each goal, so an individual rollout may visit fewer states than a less efficient pre-trained policy would. Our method partially makes up for this by prioritizing far-away goals with long paths and by favoring a set of goals with disjoint paths in one batch by maximizing occupancy. Even so, one possible direction of exploration to yield more informative exploration behaviors at the cost of algorithmic complexity would be to pre-train behaviors that explicitly maximize coverage.